Almeida P., Novais P., Neves J., Prediction Tools for Student Learning Assessment in Professional Schools, in Proceedings of the ESM 2008 - The 22nd annual European Simulation and Modelling Conference, Le Havre, France, October, ISBN 978-90-77381-44-1, pp 121-129, 2008.

PREDICTION TOOLS FOR STUDENT LEARNING ASSESSMENT IN PROFESSIONAL SCHOOLS

Paulo Almeida[1], Paulo Novais[2] and José Neves[2]

[1] Centro de Formação Profissional da Indústria de Calçado, São João da Madeira, Portugal
paulo@cfpic.pt

[2] Departamento de Informática-CCTC The University of Minho, Braga, Portugal
{pjon, jneves}@di.uminho.pt

KEYWORDS
Artificial Intelligence, Multi-Valued Extended Logic Programming, Quality-of-Information, MOODLE.

ABSTRACT

Professional Schools are in need to access technologies and tools that allow the monitoring of a student evolution course, in acquiring a given skill. Furthermore, they need to be able to predict the presentation of the students on a course before they actually sign up, to either provide them with the extra skills required to succeed, or to adapt the course to the students' level of knowledge.
Based on a knowledge base of student features, the Student Model, a Student Prediction System must be able to produce estimates on whether a student will succeed on a particular course. This tool must rely on a formal methodology for problem solving to estimate a measure of the quality-of-information that branches out from students' profiles, before trying to guess their likelihood of success.
Indeed, this paper presents an approach to design a Student Prediction System, which is, in fact, a reasoner, in the sense that, presented with a new problem description (a student outline) it produces a solved problem, i.e., a diagnostic of the student potential of success.

INTRODUCTION

This work presents an approach to the design of a Student Prediction System (SPS), with a strong emphasis on its reasoning module, endorsed by a process of quantification of the Quality-of-Information (QI) that stems out from the analysis of the student data, here understood in terms of the extensions of the predicates or logical functions that make it. It is intended to produce a Decision Support System (DSS) to help teachers, tutors, psychologists and others, to forecast problems on the evolution of the student state of knowledge, and decide and take the appropriate measures to work them out, on time.
The SPS uses data from several sources to build and evolve the student models. Based on those models it presents the decision makers with a forecast of the student learning success. We start by summarizing the notion of e-Learning Systems and present some features of the MOODLE e-Learning one. Next, we present a methodology for representing data and knowledge in SPS, and evaluate the QI of a student model, as a way to assess the reliability of the information it contains. Finally, we further elaborate on the concept of student models and system beliefs, and our vision of a SPS, including its Student Model Builder and, mainly, its Student Prediction Module.

LEARNING MANAGEMENT SYSTEMS AND MOODLE

Learning Management Systems/Course Management Systems (LMS/CMS) are domain independent, general purpose programmes/platforms, which provide authoring, sequencing, and aggregation tools that structure content, to ease the learning process. It is the duty of the course designer to select and organize the matter in order to build a tutorial module for a given knowledge domain. MOODLE platform (Moodle - A Free, Open Source Course Management System for Online Learning, n.d.) is an instance of a LMS/CMS.

The MOODLE Learning Management System

MOODLE (Modular Object-Oriented Dynamic Learning Environment) has a number of interactive learning activity components like forums, chats, quizzes and assignments. Very interesting is the lesson activity, wherein it is possible to write interactive learning tasks with conditional paths, adapting to the student learning process. In addition, MOODLE includes a logging module to track users' accesses and the activities and resources that have been accessed. Administrators and teachers can extract reports from this data. Figure 1 presents a high level view of the MOODLE modules.

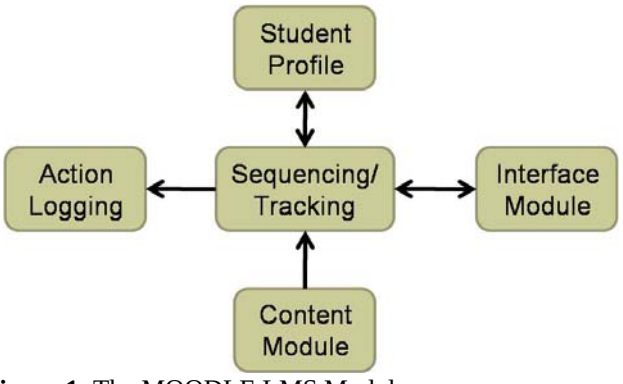

Figure 1: The MOODLE LMS Modules

Almeida P., Novais P., Neves J., Prediction Tools for Student Learning Assessment in Professional Schools, in Proceedings of the ESM 2008 - The 22nd annual European Simulation and Modelling Conference, Le Havre, France, October, ISBN 978-90-77381-44-1, pp 121-129, 2008.

LMSs should have some sort of knowledge about the students and about their learning processes. This knowledge, i.e., the beliefs the system has about the students' state of knowledge, is commonly called the Student Model (SM). Indeed, without a SM, the system would simply behave the same way for all students.

KNOWLEDGE REPRESENTATION FOR INFORMATION QUALITY ASSESSMENT

Knowledge representation is a crucial factor regarding the success of the operation of a DSS (Neves, 1984; Way, 1991; Analide et al., 2006: 436-442; Ginsberg, 1991).
A suitable representation of incomplete information and uncertainty is needed, one that supports non-monotonic reasoning.
In a classical logical theory, the proof of a theorem results on a *true* or *false* truth value, or in an *unknown* value. On contrary, in a Logic Program (LP), the answer to a question is given by the logical constants *true* or *false*. This is a consequence of the limitations of the knowledge representation in a LP, because explicit representation of negative information is not allowed. Additionally, the operational semantics applies the Closed-World Assumption (CWA) (Hustadt, 1994) to all predicates. Usually, LP represents implicitly negative information assuming the application of reasoning according to the CWA.
An extended logic program (Program 1), on the other hand, is a finite collection of rules of the form (Neves, 1984; Gelfond and Lifschitz, 1990):

$$q \leftarrow p_1 \wedge \ldots \wedge p_m \wedge \text{not } p_{m+1} \wedge \ldots \wedge \text{not } p_{m+n} \quad (1)$$
$$? \, p_1 \wedge \ldots \wedge p_m \wedge \text{not } p_{m+1} \wedge \ldots \wedge \text{not } p_{m+n} \quad (2)$$

Program 1: An Extended Logic Program

where ? is a domain atom denoting falsity, p_i and q are classical ground literals, i.e., either positive atoms or atoms preceded by the classical negation sign ¬. Every program is associated with a set of abducibles. Abducibles can be seen as hypotheses that provide possible solutions or explanations for given queries, here given in the form of exceptions to the extensions of the predicates that make the program.
The objective is to provide expressive power for representing explicitly negative information, as well as directly describe the CWA for some predicates, also known as predicate circumscription (Parsons, 1996). Three types of answers to a given question are then possible: *true*, *false* and unknown (Neves, 1984). The representation of null values will be scoped by Extended Logic Programming (ELP). We will consider two types of null values: the first will allow for the representation of unknown values, not necessarily from a given set of values; and the second will represent unknown values from a given set of possible values. To see how null values can be used to represent unknown information, let us consider the extensions of some predicates whose attributes resemble that of a SM, namely:

```
had_attended: Student x StrValue
motivation: Student x Value
grade_PA: Student x Value
```

where the former argument denotes a given student and the second represents the value of a particular asset (e.g., `motivation(ana, 1)` means that the motivation of the student Ana has the logical value 1).
In Program 2, the symbol ¬ denotes strong negation, symbolizing what should be interpreted as false, and the term *not* designates negation-by-failure.

$$\text{motivation(ana,1)}$$
$$\neg\text{motivation}(S, V) \leftarrow \text{not motivation}(S, V)$$

Program 2: An extension of the predicate that denotes the motivation of student Ana

Let us admit that the motivation of another student, say, Diana, has not yet been established. This will be denoted by a null value of the type *unknown*, as is given in Program 3: the student has some motivation but it is not possible to be certain about its *truth* value. In the first clause, the symbol $\perp$ represents a null value of an undefined type. It is a representation that assumes any value as a viable solution, but without being given a clue about which value one is speaking about. It is not possible to compute the value of the motivation of student Diana. The third clause of the program (the closure of predicate motivation) discards the possibility of being assumed as *false* any question on the specific value of motivation for Diana.

$$\text{motivation(diana}, \perp)$$
$$\neg\text{motivation}(S, V) \leftarrow \text{not motivation}(S, V), \text{not exception(motivation}(S, V))$$
$$\text{exception(motivation}(S, V)) \leftarrow \text{motivation}(S, \perp)$$

Program 3: Motivation of student Diana, with an *unknown* value.

Let us now consider the case in which the value of the motivation for a certain student is foreseen to be 0.60, with an error margin of 0.15. It is not possible to be positive, concerning the motivation truth value. However, it is false that the student has a motivation value of 0.80 or 1. This example suggests that the lack of knowledge may only be associated to an enumerated set of possible known values. As a different case, let us consider the motivation of the student Paulo, that is unknown, but one knows that it is specifically 0.30 or 0.50 (Program 4).

$$\neg\text{motivation}(S, V) \leftarrow \text{not motivation}(S, V), \text{not exception(motivation}(S, V))$$
$$\text{exception(motivation}(S, V)) \leftarrow \text{motivation}(S, \perp)$$
$$\text{motivation(ana,1)}$$
$$\text{motivation(diana}, \perp)$$
$$\text{exception(motivation(carlos}, V)) \leftarrow V \geq 0.45 \wedge V \geq 0.75$$
$$\text{exception(motivation(paulo,0.30))}$$
$$\text{exception(motivation(paulo,0.50))}$$

Program 4: A logical illustration of the motivations for students Ana, Diana, Carlos and Paulo

Using ELP, as the logic programming language, a procedure given in terms of the extension of a predicate called *demo*, is given by Program 5. This predicate allows one to reason about the body of knowledge presented in a particular

domain, set on the formalism referred to above. Given a question, it returns a solution based on a set of assumptions. This meta-predicate is defined as:
Demo: Question x Answer
where Question denotes a theorem to be proved and Answer denotes a truth value: *True* (*T*), *False* (*F*) or *Unknown* (*U*).

$$demo(Q, T) \leftarrow Q$$
$$demo(Q, F) \leftarrow \neg Q$$
$$demo(Q, U) \leftarrow \text{not } Q \wedge \text{not } \neg Q$$

Program 5: An extension of the meta-predicate *demo.*

QUALITY-OF-INFORMATION OF THE STUDENT MODELS

Until now, we have seen that SM may not always produce predictions based only on LP representations of system beliefs. We have also seen how SM uses ELP to express its uncertainty and overcome this limitation. In any decision making process, the decision is made without having all the information pertaining to the problem. When the decision maker has to, he/she makes the decision using the available information, to the best of his/her knowledge. How much a teacher relies on the predictions at hand? How can SM provide him/her with a measure of the quality of that information?
Let i (i ∈ 1 … m) represent the predicates whose extensions make an extended logic program that models the universe of discourse, and j (j ∈ 1 … n) the attributes of those predicates. Let $x_j \in [min_j, max_j]$ be a value for attribute j. To each predicate is also associated a scoring function, $V_{ij}[min_j, max_j] \rightarrow 0 \ldots 1$, that gives the score that predicate i assigns to a value of attribute j in the range of its acceptable values, i.e., its domain (for simplicity, scores are kept in the interval [0 … 1]), here given in the form:

all(attribute_exception_list, sub_expression, invariants)

This denotes that *sub_expression* should hold for each combination of the exceptions of the extensions of the predicates that represent the attributes in the *attribute_exception_list* and the invariants.
This is further translated by introducing three new predicates. The first predicate creates a list of all possible exception combinations (pairs, triples, ..., n-tuples) as a list of sets determined by the domain size (and the invariants). The second predicate recourses through this list and makes a call to the third predicate for each exception combination. The third predicate denotes *sub_expression*, giving, for each predicate, the respective score function. The Quality-of-Information (QI) with respect to a generic predicate P is therefore given by QIP = 1/Card, where Card denotes the cardinality of the exception set for P, if the exception set is not disjoint. If the exception set is disjoint, the quality of information is given by:

$$QI_P = \frac{1}{C_1^{Card} + \cdots + C_{Card}^{Card}}$$

where C_{Card}^{Card} is a card-combination subset, with *Card* elements.
The next element of the model to be considered is the relative importance that a predicate assigns to each of its attributes under observation: w_{ij} stands for the relevance of attribute *j* for predicate *i* (it is also assumed that the weights of all predicates are normalized), i.e.:

$$\forall i \sum_{j=1}^{n} w_{ij} = 1$$

It is now possible to define a predicate's scoring function, i.e., a value $x = (x_1 \ldots x_n)$ in the multi-dimensional space defined by the attributes domains, which is given in the form:

$$V_i(x) = \sum_{j=1}^{n} w_{ij} * V_{ij}(x_j)$$

It is therefore possible to measure the QI that occurs as a result of a logic program that makes a SM, by posting the $V_i(x)$ values into a multi-dimensional space and projecting it onto a two-dimensional one.
Using this procedure, one gets (Figure 2).

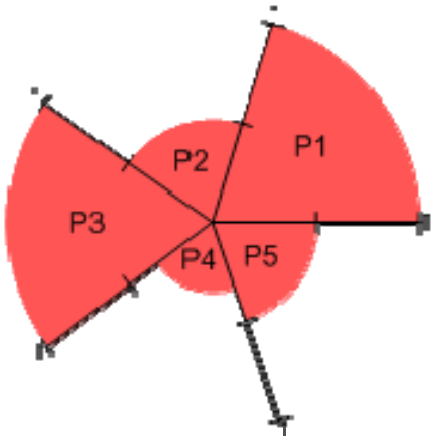

Figure 2: A measure of the QI for a Logic Program or Theory P

where the dashed n-slices of the circle (built on the extensions of the predicates $p_1 \ldots p_5$) denote the QI that is associated with each of the predicate extensions that make the logic program. It is now possible to evaluate the QI of the SMs of Ana and Diana (Figures 3 and 4). Let us consider the logic programs 6 and 7, which represent a set of beliefs about students Ana and Diana, as well as exceptions to those beliefs, which are given in the form:

$$motivation(ana, 1)$$
$$had\_attended(ana, geometry)$$
$$\neg grade\_PA(S, V) \leftarrow \text{not } grade\_PA(S, V), \text{not } exception(grade\_PA(S, V))$$
$$exception(grade\_PA(ana, 14))$$
$$exception(grade\_PA(ana, 16))$$

Program 6: A formal description of the Universe of Discourse for Ana SM

Almeida P., Novais P., Neves J., Prediction Tools for Student Learning Assessment in Professional Schools, in Proceedings of the ESM 2008 - The 22nd annual European Simulation and Modelling Conference, Le Havre, France, October, ISBN 978-90-77381-44-1, pp 121-129, 2008.

$\neg$motivation(S, V) $\leftarrow$ not motivation(S, V),
not exception(motivation(S, V))
exception(motivation(S, V)) $\leftarrow$ motivation(S, $\perp$)
$\neg$had_attended (S, V) $\leftarrow$ not had_attended(S, V),
not exception(had_attended(S, V))
exception(had_attended (S, V)) $\leftarrow$ had_attended(S, $\perp$)
$\neg$grade_PA(S, V) $\leftarrow$ not grade_PA (S, V),
not exception(grade_PA (S, V))
motivation(diana, $\perp$)
had_attended(diana, $\perp$)
exception(grade_PA(diana, 8))
exception(grade_PA(diana, 11))

Program 7: A formal description of the Universe of Discourse for Diana SM

In order to find the relationships among the extensions of these predicates, we evaluate the relevance of the QI, given in the form $V_{motivation}$(ana)= 1; V_{grade_PA}(ana)= 0.5; $V_{had\_attended}$(ana)= 1. Roughly, this means that we are sure about the motivation and attendance information of Ana; but we are not so sure about the information on the percentage of right answers. As for Diana, we are not sure whatsoever about her motivation and attendance, although we have some assurance (0.5) on her percentage of right answers.

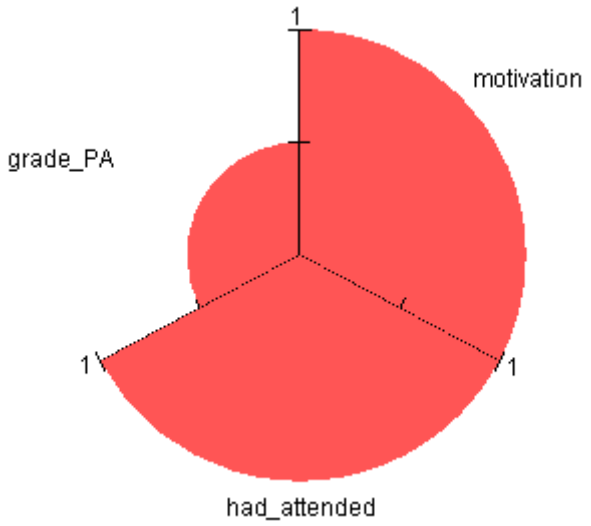

Figure 3: A measure of the Quality-of-Information for Ana

STUDENT MODELS

Therefore, a SM may be defined as a representation of the set of beliefs that a system has about a student (Self, 1994). We will follow Self's definition but we will use Multi-Valued Extended Logic Programs (MVELP) to express those beliefs, being the truth values bound to a proven theorem given in terms of a composition of the measures of the Quality-of-Information of the predicates that make it (Neves, et al 2007). Indeed, let B_Ap denote that a program A subscribes the substance (i.e., the essence of the extension) of predicate p.

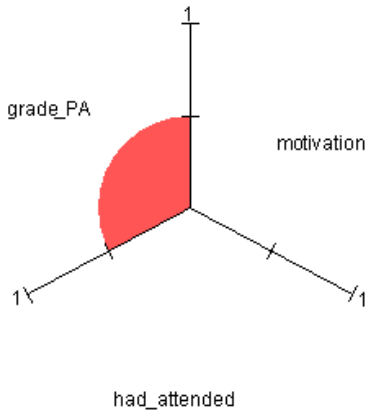

Figure 4: A measure of the Quality-of-Information for Diana

The belief set B_A configures the extensions of the set of predicates assumed by program A: B_A= {p | B_Ap}. By applying this line of thought, we can define:

- B_S as the student set of beliefs;
- B_C as the computer system set of beliefs. This set includes the extensions of those predicates that the system believes with respect to the general student behaviour; and
- SM as the subset of the system's beliefs which stand for the beliefs that the system C has about the student S: B_C(S)= {p | B_Cp(S)}.

The student's beliefs are not known by the system; therefore, all reasoning about the student has to be made through the analysis of the SM. As for B_C, this set contains the domain knowledge, beliefs about student behaviour, as well as SMs, among others sources of information. The relationship between these sets is shown in Figure 5.

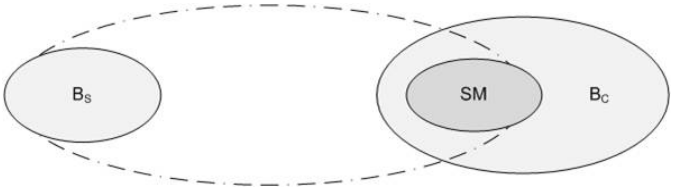

Figure 5: Relationship between B_S, B_C and SM (Self, 1994)

MOODLE does not have a true SM. It only has a student profile. However, MOODLE does collect metrics about all sorts of actions made by the users. SMs can be built from the history logs of the platform and updated with student activity logs, as we shall see later. In fact, there is a great amount of discussion about the feasibility of SMs from student interactions with learning platforms. Arroyo et al. (2004: 782-784), Johnson et al. (2005) and Mislevy et al. (1999: 437-446), have some work done on this area, mainly through the use of Bayesian Networks in conjunction with Data Mining, to model students' behaviour.

Almeida P., Novais P., Neves J., Prediction Tools for Student Learning Assessment in Professional Schools, in Proceedings of the ESM 2008 - The 22nd annual European Simulation and Modelling Conference, Le Havre, France, October, ISBN 978-90-77381-44-1, pp 121-129, 2008.

STUDENT PREDICTION SYSTEM

A SPS configures a DSS made of two modules: a module to create and update the SMs and the system beliefs, B_C; and a module to estimate the possibilities of success of the student on a course. The former is the Student Model Builder (SMB); the latter is the Student Prediction Module (SPM). Its purpose is to help teachers and others to predict success or failure on the learning path of a particular student, on a given course, and to take the proper measures to avoid failure. The architecture of SPS is depicted in Figure 6.

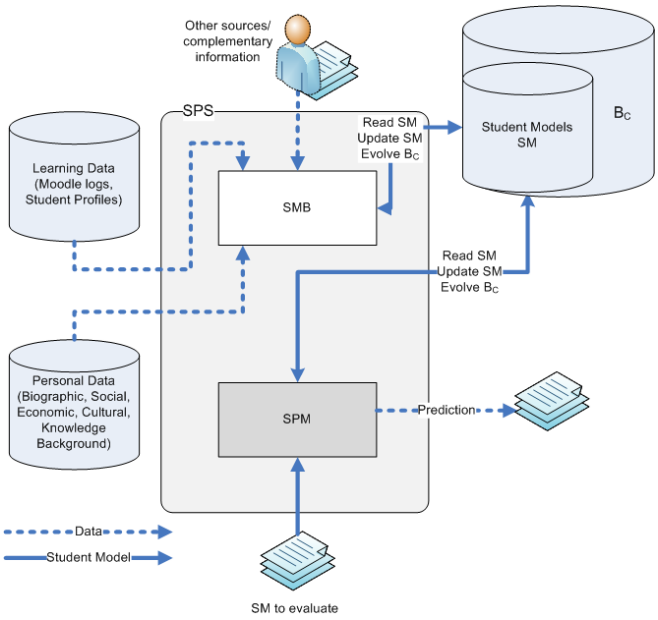

Figure 6: The Student Prediction System Architecture

BUILDING SYSTEM BELIEFS AND STUDENT MODELS FROM MOODLE

In order to be able to produce SMs, the computer system must have some beliefs about the student behaviour, face to the pedagogical resources from a given domain. These beliefs, as we had already seen, are integrated into B_C.
MOODLE has an activity logger to register users' accesses (i.e., user ID, IP and time of access) and the activities and resources that have been approached. From the log, MOODLE is able to generate, for each student, activity reports. This information can be combined with biographical, socio-economic and cultural data, as well as former academic history, in order to obtain a more complete representation of the students' evolution. Figure 7 configures a simplified information model of MOODLE activity log.

Extracting System Beliefs B_C

It is now possible, using data mine techniques for mining association rules from databases, in a way similar to the one that was exploited by Lukichev, Diaconescu and Giurca (2007), which is in itself based on the work of Agrawal, Imielinski and Swami(1993), and Agrawal and Srikant (1994), to derive a set of beliefs from MOODLE logs of student transactions, to establish B_C.

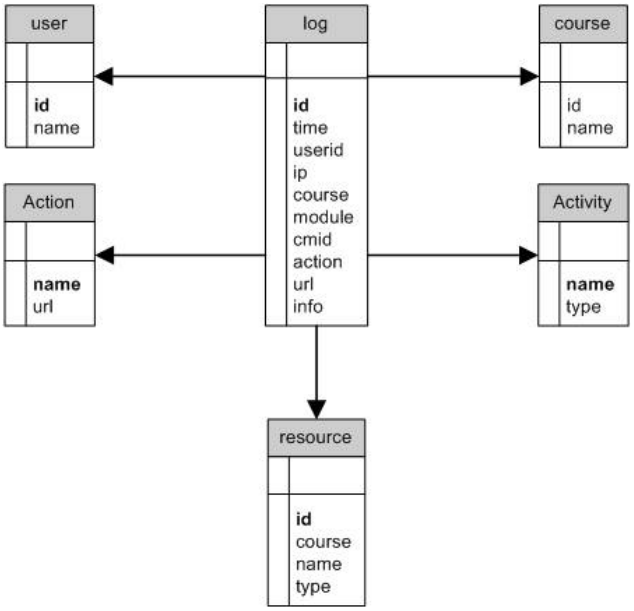

Figure 7: Excerpt from MOODLE Information Model

Generally, the idea is to find patterns, correlations and association rules on data sets of the student interactions with MOODLE. The discovering of association rules is used to come across elements that occur together in data sets, with a given confidence and support values. These values may now be used to establish an order relation on the set of beliefs.

$$
B_C = \begin{cases}
\ldots \\
\text{motivaton}(S,M) \leftarrow M \text{ is num\_visits}(S) * \text{avg\_visit\_time}(S) \\
\text{course\_requirements}(\text{shoe\_design}, \text{drawing}) \\
\text{lesson\_success}(S, \text{shoe\_cad\_SW}) \leftarrow \\
\quad \text{num\_lesson\_visits}(S, \text{shoe\_cad\_SW}, N) \wedge \\
\quad \text{avg\_lesson\_time}(S, \text{shoe\_cad\_SW}, T) \wedge N * T \geq 1600 \\
\text{course\_grade}(S, \text{shoe\_design}) \leftarrow \\
\quad \text{num\_course\_visits}(S, \text{shoe\_design}, N) \wedge \\
\quad \text{avg\_course\_time}(S, \text{shoe\_design}, T) \wedge \\
\quad V \text{ is } N*T \wedge V \geq 3000 \\
\text{grade\_PA}(S, \text{GPA}) \leftarrow \text{GPA is } \sum_{i=1}^{n} \text{grade\_P}(S, C_i, GP_i)/n \\
\ldots
\end{cases}
$$

Program 8: Extract of System Beliefs B_C

We use Rule Based Programming (RBP) (Kowalski and Levy, 1996), here given in terms of productions of the Multi-Valued Extended Logic Programming (MVELP) language. In RBP, given a rule in the form Q ← P, Q is triggered whenever P occurs. As an example, we may express association rules in terms of a MVELP program (Program 8).
In this example, the system believes:

- That *motivation* is a function of the number of visits to the platform, plus the average time per visit;
- That it is required to attend the *drawing* course before attend *shoe_design* course;
- That high number of visits to lesson *shoe_cad_SW* and high average time spent with its pedagogical resources, usually predicts success on that lesson;

Almeida P., Novais P., Neves J., Prediction Tools for Student Learning Assessment in Professional Schools, in Proceedings of the ESM 2008 - The 22nd annual European Simulation and Modelling Conference, Le Havre, France, October, ISBN 978-90-77381-44-1, pp 121-129, 2008.

- That some number of visits to *shoe_design* plus some average course visit time, when above 3000, probably means that the student will grade on that course;
- That grade point average is the mean of the grade points of all attended courses.

Systems beliefs can also be added or edited manually by teachers, tutors, psychologists or others (e.g., *motivation*, *course_requirements* and *grade_PA*).

Building and Updating Student Models

The SM can be initialized through a combination of system default assumptions and the substance of inquiries presented to the students by the time of their enrolment on a course, being updated through the student interactions with the system. Besides, there is information we can not derive from log files (e.g., school attitude or socio-economical status). Indeed, this information must come from other sources.

$$SM_{Ana} = \begin{cases} \text{had\_attended(ana, drawing)} \\ \text{num\_visits(ana,20)} \\ \text{avg\_visit\_time(ana,200)} \\ \text{num\_course\_visits(ana, shoe\_design,16)} \\ \text{avg\_course\_time(ana, shoe\_design,200)} \\ \text{grade\_P(ana, drawing,16)} \end{cases}$$

Figure 8: SM for Ana

Let us consider the following subset of MOODLE metrics, for a given student, attending a given course:

num_visits: number of visits to the Moodle platform;
avg_visit_time: average time per visit;
num_course_visits: number of visits to the resources of a course; and
avg_course_time: average time spent per course visit.

For example, the student Ana, attending a *shoe_design* course, may have the following SM (Figure 8). This logic program stands for the facts that the system knows about the student. The system beliefs about student behaviour have not yet been applied to the SM of Ana.

In fact, once the student model has been initialized, there are two sources of information on the basis of which the student model may be updated: the student's inputs to MOODLE, and the current contents of B_C.

$$SM_{Ana} = \begin{cases} \text{had\_attended(ana, drawing)} \\ \text{num\_visits(ana,20)} \\ \text{avg\_visit\_time(ana,200)} \\ \text{num\_course\_visits(ana, shoe\_design,16)} \\ \text{avg\_course\_time(ana, shoe\_design,200)} \\ \text{grade\_P(ana, drawing,16)} \\ \text{motivation(ana,4000)} \\ \text{course\_grade(ana, shoe\_design)} \\ \text{grade\_PA(ana,16)} \end{cases}$$

Figure 9: SM after applying the System Beliefs, B_C

Generally, the SMB gets the SM of a given student, analyses the SM, evaluates the quality of SM's information and updates the SM with its beliefs about the student. These beliefs result from instantiating the system set of beliefs, B_C, to this particular student.

For instance, it is possible to produce these pinpointings for Ana (Figure 9).

We can see that the system updated the SMs with instantiations of its beliefs about *motivation*, *course_grade* and *grade_PA*. These beliefs stand for the information given by SMB.

When the SMB encounters evidence that the current SM is inaccurate, for example, by observing that the student acts differently to the way the SM would predict, namely due to higher number of visits to a course, more time spent with resources, higher percentage of right answers, then the SMB may try to find and adjust those components necessary to enable the model to correspond to the observed student behaviour (this has implicit a time dimension not considered in this work).

STUDENT PREDICTION MODULE

In order to predict the success of the students on a given course, we must learn what the best predictors of success on that course are. In the social sciences, multiple regression procedures are very widely used in research. In our work it is assumed that those predictors (as well as its coefficients) are already known, and are a subset of those used by Kruck and Lending (2003). For example, let us consider the following attributes of the SM that are predictors of success on a course *C* (with a certain correlation coefficient R):

motivation, *M*: real
had_attended, *H*: {0 (false), 1 (true)}
grade_PA, *GPA*: real

and the corresponding regression equation, or prediction model:

$$Y = a + b_1 \times M + b_2 \times H + b_3 \times GPA$$

where *a* is the *intercept coefficient*, b_i the *slopes* or *regression coefficients* and *Y* is the grade point prediction for the student on the course.

SPM gives a prediction of student success on *C* by displaying that of a student that attended the course in the past, which is akin to the applicant being evaluated (in education, past behaviour is determined to be the best predictor of future behaviour (Aleamoni, 1977)).

Indeed, and in terms of implementation, we draw on Case Based Reasoning (CBR) (Aamodt and Plaza, 1994), whose knowledge base of past cases is given by B_C. Generally, the reasoner (Figure 10) operates in the form:

- Find the set of SMs of students that attended *C*, before having attended the course. Then retrieve those similar to the SM being analyzed, i.e., SM_a. Our approach is to use the prediction model itself to compute the similarity. Roughly speaking, the retrieved set is $\{SM_i\}$ such as $Y(SM_i) \sim Y(SM_a)$. Complementarily, the reasoner may also use system beliefs to fine grain the selection.

- Reuse the SM_i that best matches SM_a to estimate the probability of success. Label SM_a with that probability.
- Revise SM_i. Assess the grade point of SM_i after attending *C*: does it confirm the prediction? At this stage, a teacher should decide measures to remedy possible lacks of knowledge.
- Retain the prediction and the measures decided to avoid failure together with SM_a.

A well defined reasoner must not only solve the case but also explain the solution. The student must be informed and confronted with the predictions and the measures decided for him/her.

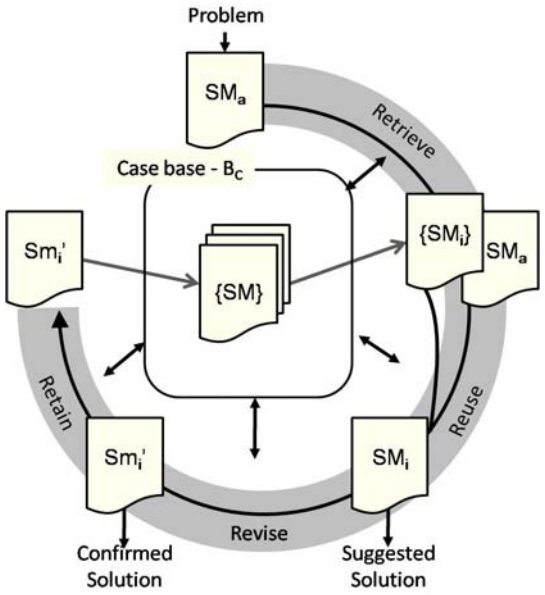

Figure 10: Case Based Reasoner (Aamodt and Plaza, 1994)

The framework just presented above is a theoretical model of a SPS. The development is at an early stage. For now, in order to achieve this first implementation, we favour the simplicity of the multiple regression prediction model and that of the reasoning process, upon more complex, though more reliable, prediction models and reasoning processes. After the first version and the analysis of its results, we aim to evolve the system, introducing a more reliable prediction model and a more accurate reasoning process.

CONCLUSIONS

The particular nature of the learning process demands grounded decisions taken on time, to be able to detect problems when they arise and avoid failures. A SPS, as the one described here, is able to build and evolve SMs from the logs of an e-Learning platform. It is also able to output predictions of student learning paths, with a given Quality-of-Information, using CBR and a multiple regression prediction model. The Quality-of-Information is critical to assure the credibility of the whole process of forecast and, therefore, the verdict of the decision makers.

There is much work to be done on the prediction model. Bayesian Networks are better fitted to model the conditional relationships between SM's predictor attributes. There is also work to do on behalf of improvements on the representation of cases (SM) and on the overall reasoning process. We are currently working on these issues.

Finally, we are putting up a prototype of a Student Assessment System (Almeida, 2008) which includes a SPS similar to the one we have just described.

ACKNOWLEDGEMENTS

We would like to thanks the Professional School *Centro de Formação Profissional da Indústria de Calçado, São João da Madeira, Portugal*, for the kindness in letting us to use its MOODLE e-Learning Platform in this work.

REFERENCES

Aamodt, A. and Plaza, E. (1994) "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches", AI Communications, Vol. 7 Nr. 1, pp 39-59, March 1994.

Agrawal R., Imielinski T., Swami A. (1993) "Mining association rules between sets of items in large databases", Buneman and Jajodia, editors, Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 207–216, Washington, D.C., USA

Agrawal R., Srikant R. (1994) 'Fast algorithms for mining association rules', Bocca, Jarke, and Zaniolo, editors, Procedures of the 20th International Conference on Very Large Data Bases, VLDB, pp. 487–499, Morgan Kaufmann Publishers, Los Altos, California, USA.

Aleamoni, L. M. (1977) "Can grade point average be more accurately predicted?" Journal of Educational Psychology, 69, 225-227.

Almeida, P. (2008) Tools for Student Learning Assessment in Professional Schools, MSc Project, University of Minho, Braga, Portugal.

Analide C., Novais P., Machado J., Neves J. (2006) "Quality of Knowledge in Virtual Entities", Encyclopaedia of Communities of Practice in Information and Knowledge Management, Coakes and Clarke, editors, Idea Group Reference, pp. 436-442.

Arroyo, I., Murray, T., Woolf, B. and Beal C. (2004) "Inferring unobservable learning variables from students' help seeking behaviour", Workshop Proceedings of International Conference on Intelligent Tutoring Systems, pp. 782-784.

Gelfond, M. and Lifschitz V. (1990) "Logic Programs with Classical Negation", in Proceedings of the International Conference on Logic Programming.

Ginsberg, M.L. (1991) Readings in Non-monotonic Reasoning, Morgan Kauffman Publishers, Los Altos, California, EUA.

Hustadt, U. (1994) "Do we need the closed-world assumption in knowledge representation?" in Working Notes of the KI'94 Workshop, Saarbrüken, Germany.

Jonsson, A., Johns, J., Mehranian, H., Arroyo, I., Woolf, B., Barto, A., Fisher, D. and Mahadevan, S. (2005) "Evaluating the feasibility of learning student models from data", Proceedings of the Workshop on Educational Data Mining at Association for the Advancement of Artificial Intelligence (AAAI).

Kowalski, T. and Levy, L. (1996) Rule-Based Programming, Springer.

Kruck, S. and Lending, D. (2003) "Predicting Academic Performance in an Introductory College-Level IS Course", Information Technology, Learning, and Performance Journal, Vol. 21, No. 2, Fall 2003.

Lukichev S., Diaconescu M. and Giurca A. (2007) "Empowering Moodle with Rules and Semantics", Proceedings of the European Semantic Web Conference, Innsbruck, Austria, May 30.

Mislevy, R.J., Almond, R.G., Yan, D., and Steinberg, L.S. (1999) "Bayes nets in educational assessment: Where do the numbers come from?", Laskey and Prade, editors, Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, pp. 437-446.

Moodle - A Free, Open Source Course Management System for Online Learning, [Online], Available: http://moodle.org/ [10 Jul 2008].

Neves, J. (1984) "A Logic Interpreter to Handle Time and Negation in Logic Data Bases", in Proceedings of the ACM'84, The Fifth Generation Challenge.

Neves, J., Machado, J., Analide, C., Abelha, A., and Brito, L. (2007) "The Halt Condition in Genetic Programming", in Proceedings of the 13th Portuguese Conference on Artificial Intelligence, EPIA, Guimarães, Portugal, pp. 160-169.

Parsons, S. (1996) "Current approaches to handling imperfect information in data and knowledge bases", IEEE Transactions on Knowledge and Data Engineering, Vol. 8, No. 3, pp. 353-372.

Self, J. (1994) Formal Approaches to Student Modelling, AAI/AI-ED Technical Report No. 92.

Way, E.C. (1991) Knowledge Representation and Metaphor, Kluwer Academic Publishers, Dordrecht, Holland.

AUTHOR BIOGRAPHIES

PAULO ALMEIDA is a Systems Engineer and a MSc student at the Department of Informatics, the University of Minho, Braga, Portugal. He has been working for several years on re-engineering Professional Schools' business processes and technological infrastructures. He has also work done on the fields of systems integration and networks design and administration. He is currently the Chief Information Officer at the Professional High School Centro de Formação Profissional da Indústria de Calçado.

PAULO NOVAIS is Professor of Computer Science at the Department of Informatics at the University of Minho. He received a PhD in Computer Sciences from the same university in 2003. He is Vice-president of APPIA, the Portuguese Association for Artificial Intelligence. His current research directions span the fields of Knowledge Representation and Reasoning Systems, Multi-agent Systems, Ambient Intelligence and Collaborative Networks.

JOSÉ NEVES is Full Professor of Computer Science at the Department of Informatics at the University of Minho. He got his PhD in Computer Science from Harriet Watt University, Edinburgh, Scotland in 1984. His current research activities span the fields of Non-monotonic Logics, Knowledge Representation and Reasoning Systems and Learning.